University of San Francisco

Summarization & Sentiment Analysis for Amazon Reviews

Written by

Goal Diggers

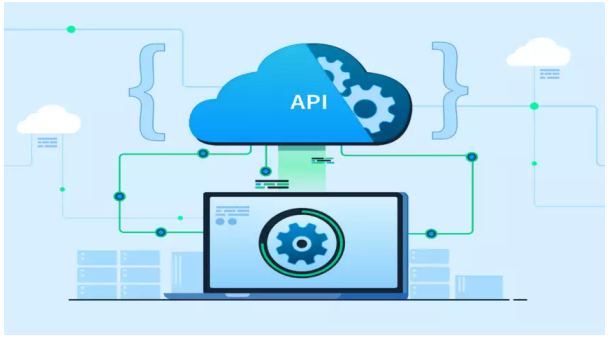
—

Akul Bajaj
Andres Martinez Tobon
Connor MacMillan
Manish Kumar Vuppugandla
Patricia Ornelas Jauregui

Problem Statement

Online shopping can be difficult, especially for clothes, because there are so many options and the product descriptions are often vague or inaccurate. Online marketplaces like Amazon have customer reviews that tell buyers about the fit, quality, and design of fashion items so they can make better decisions. Despite all this, reading through several evaluations can be time-consuming and boring, leading to decision fatigue and doubt — so much so that customers end up abandoning the effort altogether. One potential answer to this issue is to develop summaries of fashion items based on their reviews, which can provide an overview of major characteristics and opinions at a glance. It is unknown, however, how precise and valuable these summaries are in comparison to the official product descriptions provided by the fashion manufacturers themselves. Thus, the purpose of this study is to examine the efficacy of generating summaries of fashion items from customer reviews and compare them to the official product descriptions in terms of accuracy, completeness, and customer utility, and to predict the sentiment of rating given the text in the reviews.

Dataset Overview



We used two Amazon-based datasets. One was a static dataset of Amazon reviews for different products. This Amazon review data was compiled by a UCSD PhD student, Jianmo Ni. He split the data into categories and metadata. For example, he had different datasets for the review data and metadata of categories like books, software, video games, etc. We specifically used the Amazon fashion reviews because Goal Diggers are very fashionable people. This was a very large dataset, as it has 883,636 reviews.

Our second dataset was also from Amazon. This dataset we scraped ourselves from Amazon using the Rapid API software. We chose 39 different fashion based items from Amazon that

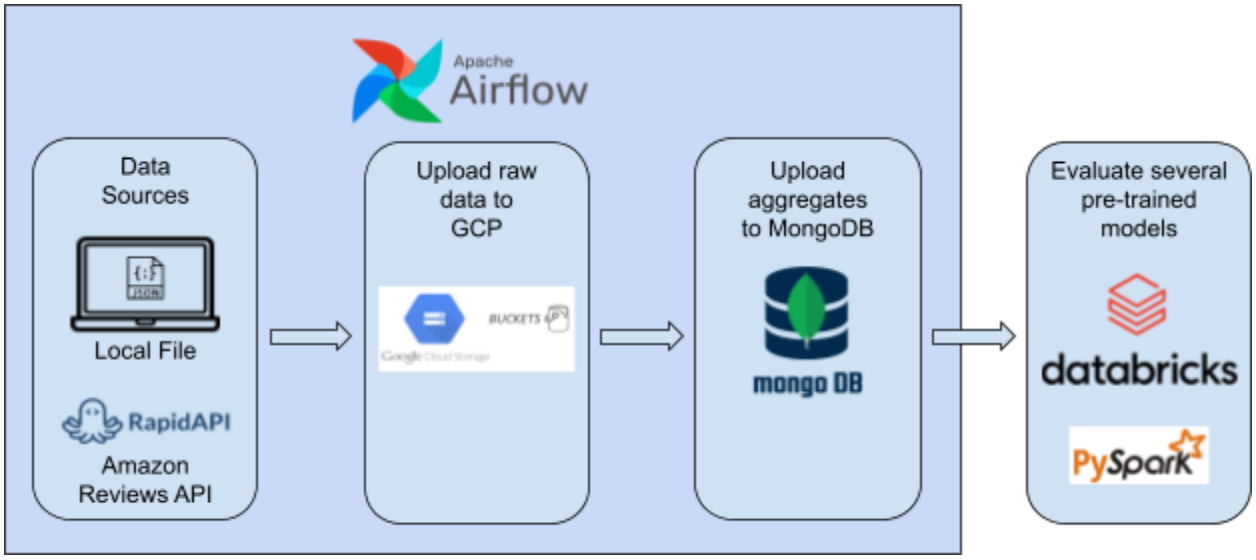
appeared to have several reviews. We used each item's ASIN number, unique to the product on Amazon, to scrape the review data.

Finally, we combined the two datasets, but there were many more data points in the static dataset, than in the scraped API datasets, so the static data makes up a majority of the merged collection. The final columns we have are ASIN, Review, Rating, and Title for over 200,000 products.

Analytic Goals

Our goal with this project is to create product summaries using reviews and performing sentiment analysis. We will compare different models to determine the best option for this use case.

Data Engineering Pipeline



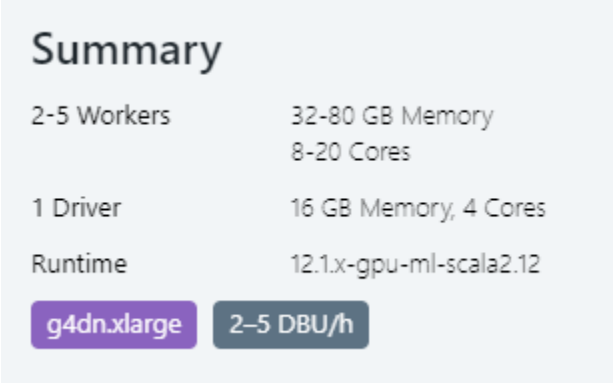
Given our data sources, we used Airflow to upload this raw data to GCP storage. Then, we had a step in our DAG to read both files from GCP and create aggregates. Finally, those aggregates are uploaded to MongoDB. In Databricks, we pulled the data from MongoDB to perform our machine learning.

Preprocessing Overview

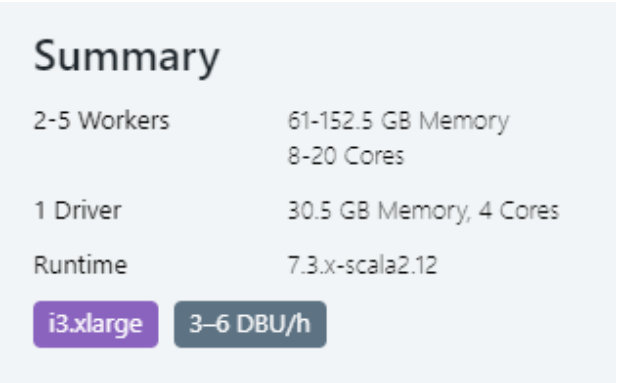
We had two distinct data sets to aggregate, one was a json file of 883,636 reviews for over 201,959 products, and the other was collected from the Amazon API. Before pushing this data to MongoDB Atlas, we needed to combine these into one document per product. Our chosen schema was to have a field for the ASIN string representing the product and a field for an array of reviews, each of which includes the score the user rated, the title of the review, and the text of

the review. After aggregating all of the reviews, we pushed them to MongoDB Atlas so that we could pull it into our DataBricks notebooks.

We used two different DataBricks notebooks, one on the GoalDiggers_Cluster and another on the GoalDiggers_GPU_Cluster cluster.



GoalDiggers_GPU_Cluster



GoalDiggers_Cluster

We used the GPU cluster to run the summarization model since it took roughly 40 seconds to run on the total review text, whereas the model ran in roughly 20 seconds for the same review text. Each objective had slightly different steps to preprocess the data once retrieved from MongoAtlas.

Machine Learning Overview

Supervised Sentiment analysis

SpaCy

We chose to leverage this pre-trained NLP model for sentiment analysis because of how easy it was to implement. Given some of the other models we experimented with, we wanted to see if a pre-trained model with very little overhead could result in similar or better performance. For the implementation, we take each review and pass it through an NLP function to extract keywords required to interpret meaning. Then, we predict the positive sentiment of each review using spaCy's sentiment analysis library. This returns a value between 0 and 1 that indicates the text's positivity. Finally, we divide the actual review's rating by 5 as a proxy for how positive the review was. Using spaCy, our mean absolute error between the actual rating's positivity and the predicted positivity rate was 0.315 with a runtime of 5.6 minutes. One limitation that we noticed with this library is that it does not account for neutral words, since it only returns how positive or negative the input text is.

Logistic Regression

We only used the first 95 thousand reviews in the data frame because we did not want our training function to run for too long. The resulting data is then transformed and analyzed using PyTorch. The reviews are converted into "bag-of-words" feature and the rating column is used

as the y variable, or the target variable. We trained and defined a logistic regression model using PyTorch with the loss function optimized using Adam. Finally, the trained model is evaluated using the mean absolute error (MAE).

This code took 1.30 minutes to run. We used 5 epochs and after each epoch the loss decreased substantially. In the end our MAE for logistic regression was 0.07339.

Then we showed an example to see how our model worked. For example, we tried to see what the prediction would be if the input was: “this item is a piece of junk, I hate it" and the output was far below 0.50, meaning the review would be considered bad. On the other hand, when the input was "Great! I love it!" the output was above 0.50, meaning the review would be classified as positive.

Support Vector Machines (SVM)

SVM is computationally more expensive to build and train from scratch, so instead we chose the state of the art pre-trained English language transformer-model called SiEBERT for sentiment analysis. We applied a function to compute sentiment analysis for each review in the input data. The sentiment score is normalized and adjusted to a range between 0 and 5. Then we get the error as the absolute difference between the actual rating and the predicted rating. Then we compare the ground truth to our predictions to calculate the mean absolute error (MAE). The output of the code is a data frame containing the original reviews, actual ratings, predicted ratings, and the difference between actual and predicted ratings.

Summarization by Sentiment

One of our main goals was to see how well we could summarize customer reviews of fashion items and compare them to the official product descriptions. After analyzing our data further, we instead saw an opportunity to combine the sentiment analysis and content summarization tasks into a pipeline that breaks down the overall positive, negative, and neutral sentiment of each product into easy-to-read summaries.

Since this is working with summarization, we wanted to focus on the popular products that have at least 30 reviews since a buyer may not want to manually parse through too many pages of reviews. Our original data had over 200,000 products, but this went down to 1,887 popular products with a total of 241,661 individual reviews. Pulling the data from MongoDB and filtering it to only include the popular products took roughly 35 seconds.

We began by predicting the sentiment of each review and then grouping them into positive, negative, and neutral bins. The sentiment prediction step took 3.6 seconds for all of the reviews. After that, for each product, we combined all of the text corresponding to each sentiment, which took 26.14 seconds. This resulted in each product only having three sections of text corresponding to the sentiment of the reviews. To make this information digestible for buyers we then summarized all the text corresponding to each sentiment for each product using HuggingFace’s summarization pipeline. The summarization step took 12.26 minutes to

download the pretrained weights and run. This provides a concise summarization of all the sentiments for each product and hopefully provides the buyer with a general idea of all the things that previous buyers liked, disliked, and did not care for.

Example outputs for a specific product:

After Sentiment Analysis

```
 ASIN      | B01G4TEVYG
 rating    | 5.0
 review    | Perfect! Still have this : Perfect!  Still have this.
 sentiment | positive
-RECORD 13-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------
 ASIN      | B01G4TEVYG
 rating    | 5.0
 review    | Five Stars : Love it!!
 sentiment | positive
-RECORD 14-----------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
----------------------------------------
 ASIN      | B01G4TEVYG
 rating    | 2.0
 review    | Disappointed. Wish there was a photo with model to ... : Smaller than I expected. Wanted to carry iPad but fits it with lit
tle room for anything else. Disappointed. Wish there was a photo with model to shoe scale.
 sentiment | negative
```

Grouped Text by Sentiment

```
 ASIN        | B01G4TEVYG
 sentiment   | negative
 review_text | Disappointed. Wish there was a photo with model to ... : Smaller than I expected. Wanted to carry iPad but fits it with l
ittle room for anything else. Disappointed. Wish there was a photo with model to shoe scale. + Zipper Broke on Day 1. Refunded : Zipper
broke 1st day I received it. Returned fo...
-RECORD 1------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------
 ASIN        | B01G4TEVYG
 sentiment   | neutral
 review_text | Much smaller than dimensions indicate. : Measurements are not accurate.  Actual bag is approximately 10.5 inches high, 9
inches wide and depth is 3.25 inches.  I am guessing their measurements include the floral strap portion.  The two pockets do not hold e
ven a standard water bottle.  I will be ...
-RECORD 2------------------------------------------------------------------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------------------------------------------------------
-------------------------------------------
 ASIN        | B01G4TEVYG
 sentiment   | positive
 review_text | Durable and well-made but too small : I am so sad because I was really excited about receiving this backpack shoulder sli
ng.  This is way too small and does not measure up to the measurements in the product description. This is better suited for a small chi
ld, not an adult. This is about the size...
```

Summarized Negative Reviews (review text is truncated)

```
 ASIN         | B01G4TEVYG
 sentiment    | negative
 review_text  | Disappointed. Wish there was a photo with model to ... : Smaller than I expected. Wanted to carry iPad but fits it with
little room for anything else. Disappointed. Wish there was a photo with model to shoe scale. + Zipper Broke on Day 1. Refunded : Zipper
broke 1st day I received it. Returned fo...
 summary_text |  Zipper Broke on Day 1. Refunded : Zipper broke 1st day I received it. Returned for a refund. Disappointed. Wish there w
as a photo with model to shoe scale.
```

Lessons Learned

- We learned how to handle and wrangle large datasets in a distributed data setting, using technologies like PyMongo, PyTorch, and databricks.

- Pre-trained models can be leveraged to save time and effort in training machine learning models, especially when the data is limited or when the task at hand is well-suited for a pre-trained model.
- Training on our actual dataset performed better than using pre-trained models. For use cases where training time is not a significant factor and the dataset is small, training on the dataset would be a better option than using pre-trained models. Using pre-trained models is so easy that we would recommend trying them as a first option before training your own.

Conclusion

Results

We worked on a variety of goals, including summarizing the review, performing sentiment analysis, and building ML models with NLTK, TextRank, LSA, Logistic Regression, Support Vector machines (SVM), and deep learning models.

To build the project, we used a variety of tools and libraries, including transformers, vaderSentiment, engSpacySentiment, pandas, pyspark, pymongo, and others. The findings of the experiment demonstrated that employing NLTK resulted in effective summarization of product reviews, whilst spaCy and pretrained logistic regression demonstrated high accuracy in sentiment analysis. Deep learning models such as BERT and SiBERT also performed well.

- eng_spacysentiment - 0.315
- SieBERT - 0.795
- Linear Reg - 0.0734

Limitations

Ratings and reviews from users are valuable sources of information for consumers who are trying to decide what to buy. User ratings and reviews may not fully show how a user feels about a product because they can be affected by things like the reviewer's personal tastes, expectations, and experiences.

Some reviews may also be fake or biased, so they may not reflect what real users have said. So, relying only on ratings and reviews from other users may not give a full picture of a product's quality or suitability for a specific person's needs.

Consumers should also think about things like the product's specs, expert reviews, and the brand's or manufacturer's reputation before making a purchase. When evaluating products, it's important to get information from many different places and look at them as a whole. This will help them make decisions that meet their needs and preferences.

One of the problems with our models is that the review text might not include all of the reasons why a user gave a certain rating. So, the rating should be viewed as a proxy (rather than the absolute truth) for the user's sentiment. For instance, a user may give a garment a four-star

rating because they loved five aspects of it and disliked three others. If the reviewer only talks about the three bad things, the sentiment analysis will be negative, which goes against the fact that they gave the product four stars. Ratings alone may not be a good way to figure out how a user feels about a product. This is why the reviews were put into groups based on how they made people feel instead of how they rated the product.

References

1. https://www.datafeedwatch.com/blog/amazon-asin-number-what-is-it-and-how-do-you-get-it#:~:text=You%20can%20find%20an%20ASIN,of%20the%20product%20information%20section
2. https://medium.com/analytics-vidhya/simplifying-social-media-sentiment-analysis-using-vader-in-python-f9e6ec6fc52f
3. https://rapidapi.com/logicbuilder/api/amazon-product-reviews-keywords/pricing
4. https://nijianmo.github.io/amazon/index.html
5. https://huggingface.co/siebert/sentiment-roberta-large-english
6. https://huggingface.co/docs/transformers/main_classes/pipelines